Available online at www.sciencedirect.com
ScienceDirect
Discrete Optimization 4 (2007) 175–184
DISCRETE OPTIMIZATION
www.elsevier.com/locate/disopt

Two-machine open shop problem with controllable processing times

T.C. Edwin Cheng[a], Natalia V. Shakhlevich[b,*]

[a] *Department of Logistics, The Hong Kong Polytechnic University, Hung Hom, Kowloon, Hong Kong*
[b] *School of Computing, University of Leeds, Leeds LS2 9JT, UK*

Received 10 January 2005; received in revised form 3 August 2006; accepted 5 October 2006
Available online 19 December 2006

Abstract

We consider a two-machine open shop problem in which the job processing times are controllable and can be compressed while incurring additional costs. We show that the problem of minimizing a linear compression cost function for a given upper bound on the makespan can be solved in $O(n)$ time. For the bicriteria problem of minimizing the makespan and compression cost, we propose an $O(n \log n)$ algorithm that finds all breakpoints of the efficient frontier.

Keywords: Open shop scheduling; Controllable processing times; Knapsack problem; Resource allocation problem

1. Introduction

We consider a two-machine open shop scheduling model with controllable processing times, which is a generalization of the classical open shop model with fixed processing times. The problem can be stated as follows. A set of jobs $N = \{1, \ldots, n\}$ has to be processed by machines A and B in an arbitrary order. A usual scheduling requirement stipulates that a machine cannot process more than one job at a time, and a job is never processed on more than one machine at a time. Job preemption is not allowed. For the two operations of each job $j \in N$ that are to be processed on machines A and B, there are given "normal" processing times $\overline{a}_j$ and $\overline{b}_j$, respectively. Processing times can be compressed at the expense of consuming additional resources. The compressed processing times a_j and b_j cannot be less than the minimum processing times $\underline{a}_j$ and $\underline{b}_j$, respectively, i.e.

$$\underline{a}_j \le a_j \le \overline{a}_j \quad \text{and} \quad \underline{b}_j \le b_j \le \overline{b}_j.$$

If the actual processing times a_j and b_j are determined for all the jobs $j \in N$, then the schedule that minimizes the makespan can be found by the well-known $O(n)$-time algorithm due to Gonzalez and Sahni [6], and the optimal value of the makespan is given by

$$C_{\max} = \max\left\{\sum_{j\in N} a_j, \sum_{j\in N} b_j, \max\{a_j + b_j | j \in N\}\right\}.$$

* Corresponding author. Tel.: +44 113 343 5444; fax: +44 113 343 5468.
E-mail addresses: lgtcheng@polyu.edu.hk (T.C.E. Cheng), ns@comp.leeds.ac.uk (N.V. Shakhlevich).

1572-5286/$ - see front matter
doi:10.1016/j.disopt.2006.10.010

Compressing job j may decrease the makespan, but it implies an increase in the cost incurred for consuming additional resources. The compression cost K is expressed in terms of a penalty function, which depends on the unit compression costs α_j and β_j, and the compression amounts $x_j = \overline{a}_j - a_j$ and $y_j = \overline{b}_j - b_j$ of the two operations of job j on machines A and B:

$$K = \sum_{j \in N} \left(\alpha_j x_j + \beta_j y_j\right). \tag{1}$$

The objective is to find the optimal compression amounts x_j and y_j, $j \in N$, in order to minimize the makespan $C_{\max}$ and the compression cost K.

Prior work in the area of scheduling with controllable processing times focused on single-stage systems; see, e.g., the survey paper by Nowicki and Zdrzalka [16]. For shop models, it is known that the two-machine flow shop problem is NP-hard (see [9,16]) and the two-machine open shop problem can be reduced to linear programming (see [3]). Approximation and heuristic algorithms for flow shop and job shop problems with controllable processing times were suggested in [4,7,10–12,14,15].

In this paper we consider the two-machine case of the open shop problem and study a single criterion problem of minimizing the cost function K for a given threshold value of the makespan C, and a bicriteria problem of minimizing K and $C_{\max}$ simultaneously. Extending standard notation for scheduling problems [13], we denote the first problem by $O2|\text{contr}, C_{\max} \leq C|K$ and the second one by $O2|\text{contr}|\,(C_{\max}, K)$.

We show in Section 2 that the single criterion problem can be solved in $O(n)$ time. The properties of an optimal schedule are studied in Section 3. Based on these properties, we develop in Section 4 an algorithm that solves the bicriteria problem in $O(n \log n)$ time.

2. Single criterion problem

In this section we study the two-machine open shop problem $O2|\text{contr}, C_{\max} \leq C|K$ of minimizing the compression cost function K under the constraint that $C_{\max} \leq C$, where C is a given constant. We consider the case that $C \geq \underline{C}$, where $\underline{C}$ is the smallest possible makespan achievable if all the jobs are fully compressed:

$$\underline{C} = \max\left\{\sum_{j \in N} \underline{a}_j, \sum_{j \in N} \underline{b}_j, \max\left\{\underline{a}_j + \underline{b}_j | j \in N\right\}\right\}; \tag{2}$$

otherwise the problem makes no sense.

The algorithm by Gonzalez and Sahni [6] constructs an optimal schedule in which each machine has at most one idle interval within $[0, C_{\max}]$. In order to simplify the discussion, we introduce an artificial job $n+1$ such that the minimum processing times of both operations are equal to 0 and the maximum processing times are sufficiently large

$$\underline{a}_{n+1} = \underline{b}_{n+1} = 0, \qquad \overline{a}_{n+1} = \overline{b}_{n+1} = \sum_{j=1}^{n} \left(\overline{a}_j + \overline{b}_j\right),$$

so that job $n+1$ can fill in an idle interval on each machine in any schedule achieving this at no cost:

$$\alpha_{n+1} = \beta_{n+1} = 0.$$

Clearly, the problem with the artificial job $n+1$ is equivalent to the original one and we can limit our search to schedules that satisfy

$$\sum_{j=1}^{n+1} a_j = \sum_{j=1}^{n+1} b_j = C. \tag{3}$$

In what follows, we assume that $N = \{1, 2, \ldots, n, n+1\}$.

For the problem of minimizing the cost function K for a given value of the makespan C we introduce the following linear programming formulation with variables $u_j = a_j - \underline{a}_j$ and $v_j = b_j - \underline{b}_j$ representing the decompression

amounts of job j on machines A and B from the minimum processing times $\underline{a}_j$ and $\underline{b}_j$:

$$
\begin{aligned}
LP(AB): \text{ maximize } & \sum_{j\in N}\left(\alpha_j u_j+\beta_j v_j\right)\\
\text{subject to } & \sum_{j\in N}\left(\underline{a}_j+u_j\right)=C, && \text{(I)}\\
& \sum_{j\in N}\left(\underline{b}_j+v_j\right)=C, && \text{(II)}\\
& \left(\underline{a}_j+u_j\right)+\left(\underline{b}_j+v_j\right)\le C, \quad j\in N, && \text{(III)}\\
& 0\le u_j\le \overline{a}_j-\underline{a}_j, \quad j\in N, && \text{(IV)}\\
& 0\le v_j\le \overline{b}_j-\underline{b}_j, \quad j\in N. && \text{(V)}
\end{aligned}
$$

Constraints (I) and (II) imply that the total load of each machine equals C, condition (III) guarantees that the total processing time of any job is no larger than C, and inequalities (IV) and (V) indicate that the decompression of each operation does not exceed its upper bound.

It is easy to see that ignoring constraint (III), problem $LP(AB)$ can be decomposed into the following two continuous knapsack problems:

$$
\begin{aligned}
LP(A): \quad & \text{maximize} && \sum_{j\in N}\alpha_j u_j\\
& \text{subject to} && \sum_{j\in N}\left(\underline{a}_j+u_j\right)=C,\\
& && 0\le u_j\le \overline{a}_j-\underline{a}_j, \quad j\in N,\\
LP(B): \quad & \text{maximize} && \sum_{j\in N}\beta_j v_j\\
& \text{subject to} && \sum_{j\in N}\left(\underline{b}_j+v_j\right)=C,\\
& && 0\le v_j\le \overline{b}_j-\underline{b}_j, \quad j\in N.
\end{aligned}
$$

The solutions $\mathbf{u}^A=\left(u_1^A,\ldots,u_n^A,u_{n+1}^A\right)$ and $\mathbf{v}^B=\left(v_1^B,\ldots,v_n^B,v_{n+1}^B\right)$ to problems $LP(A)$ and $LP(B)$ can be found by the $O(n)$-time algorithm due to Balas and Zemel [1]. Clearly, if $\mathbf{u}^A$ and $\mathbf{v}^B$ do not violate constraint (III), then they determine the solution to problem $LP(AB)$. Otherwise constraint (III) is violated for exactly one job, say k, which is called *critical*. We show that in this case constraint (III) holds as an equality for job k in the solution to problem $LP(AB)$.

Theorem 1. *If for the solutions* $\mathbf{u}^A$ *and* $\mathbf{v}^B$ *to problems* $LP(A)$ *and* $LP(B)$ *constraint* (III) *does not hold and* k *is a job such that*

$$\left(\underline{a}_k+u_k^A\right)+\left(\underline{b}_k+v_k^B\right)>C, \tag{4}$$

then there exists a solution $(\mathbf{u}^*,\mathbf{v}^*)=\left(u_1^*,\ldots,u_n^*,u_{n+1}^*,v_1^*,\ldots,v_n^*,v_{n+1}^*\right)$ *to problem* $LP(AB)$ *that satisfies*

$$\left(\underline{a}_k+u_k^*\right)+\left(\underline{b}_k+v_k^*\right)=C. \tag{5}$$

Proof. Suppose k is a critical job and solution $(\widetilde{\mathbf{u}},\widetilde{\mathbf{v}})$ to problem $LP(AB)$ satisfies the inequality

$$\left(\underline{a}_k+\widetilde{u}_k\right)+\left(\underline{b}_k+\widetilde{v}_k\right)<C.$$

We demonstrate that another solution $(\mathbf{u}^*,\mathbf{v}^*)$ to problem $LP(AB)$ exists for which (5) holds and the value of the objective function $F(\mathbf{u},\mathbf{v})=\sum_{j\in N}(\alpha_j u_j+\beta_j v_j)$ of the maximization problem $LP(AB)$ at the solution $(\mathbf{u}^*,\mathbf{v}^*)$ is

no less than that at the solution $(\widetilde{\mathbf{u}}, \widetilde{\mathbf{v}})$:

$$F\left(\mathbf{u}^*, \mathbf{v}^*\right) \geq F\left(\widetilde{\mathbf{u}}, \widetilde{\mathbf{v}}\right). \tag{6}$$

Consider a linear segment in $\mathbb{R}^{2n+2}$ given by $(\mathbf{u}(\lambda), \mathbf{v}(\lambda))$ such that

$$u_j(\lambda) = \lambda \widetilde{u}_j + (1-\lambda) u_j^A,$$
$$v_j(\lambda) = \lambda \widetilde{v}_j + (1-\lambda) v_j^B,$$

$j \in N$, $0 \leq \lambda \leq 1$. The end points $\left(\mathbf{u}^A, \mathbf{v}^B\right) = (\mathbf{u}(0), \mathbf{v}(0))$ and $(\widetilde{\mathbf{u}}, \widetilde{\mathbf{v}}) = (\mathbf{u}(1), \mathbf{v}(1))$ lie in different half-spaces defined by the hyperplane $\left(\underline{a}_k + u_k\right) + \left(\underline{b}_k + v_k\right) = C$:

$$\left(\underline{a}_k + u_k(0)\right) + \left(\underline{b}_k + v_k(0)\right) > C,$$
$$\left(\underline{a}_k + u_k(1)\right) + \left(\underline{b}_k + v_k(1)\right) < C.$$

Hence there exists λ^*, $0 < \lambda^* < 1$, such that

$$\left(\underline{a}_k + u_k\left(\lambda^*\right)\right) + \left(\underline{b}_k + v_k\left(\lambda^*\right)\right) = C.$$

Define $(\mathbf{u}^*, \mathbf{v}^*) = (\mathbf{u}(\lambda^*), \mathbf{v}(\lambda^*))$. It is easy to verify that all the constraints (I)–(V) are satisfied for $(\mathbf{u}^*, \mathbf{v}^*)$, and in addition

$$F\left(\mathbf{u}^*, \mathbf{v}^*\right) = \lambda^* F\left(\widetilde{\mathbf{u}}, \widetilde{\mathbf{v}}\right) + \left(1-\lambda^*\right) F\left(\mathbf{u}^A, \mathbf{v}^B\right).$$

Since $(\widetilde{\mathbf{u}}, \widetilde{\mathbf{v}})$ is the solution to the maximization problem $LP(AB)$ while $\mathbf{u}^A$ and $\mathbf{v}^B$ are the solutions to the relaxed problems $LP(A)$ and $LP(B)$,

$$F\left(\mathbf{u}^A, \mathbf{v}^B\right) \geq F\left(\widetilde{\mathbf{u}}, \widetilde{\mathbf{v}}\right).$$

Hence (6) holds, and Theorem 1 is proved. ■

Using the condition that

$$\widetilde{u}_k + \widetilde{v}_k < u_k^* + v_k^* < u_k^A + v_k^A,$$

we formulate the following corollary that will be used in the next section to describe the structure of an optimal schedule.

Corollary 2. *For the solution* $(\mathbf{u}^*, \mathbf{v}^*)$ *to problem* $LP'(AB)$ *defined in Theorem 1 neither*

$$u_k^* = 0, \qquad v_k^* = 0 \quad (\textit{both operations of job } k \textit{ are fully compressed}),$$

nor

$$u_k^* = \overline{a}_k - \underline{a}_k, \qquad v_k^* = \overline{b}_k - \underline{b}_k \quad (\textit{both operations of job } k \textit{ are fully decompressed}).$$

It follows from Theorem 1 that, if for $\mathbf{u}^A$ and $\mathbf{v}^B$ condition (4) holds, then we can limit our search to schedules that satisfy the equality

$$\left(\underline{a}_k + u_k\right) + \left(\underline{b}_k + v_k\right) = C \tag{7}$$

in addition to the constraints of problem $LP(AB)$.

We demonstrate that problem $LP(AB)$ combined with (7) can be reduced to the following linear programming problem defined for variables u_j, v_j, $j \in N \setminus \{k\}$:

$$LP'(AB): \text{maximize} \sum_{j \in N \setminus \{k\}} \left(\alpha'_j u_j + \beta'_j v_j\right)$$
$$\text{subject to} \sum_{j \in N \setminus \{k\}} \left(\underline{a}_j + u_j\right) + \sum_{j \in N \setminus \{k\}} \left(\underline{b}_j + v_j\right) = C, \tag{I'}$$

$$\sum_{j\in N\setminus\{k\}} \left(\underline{a}_j + u_j\right) \le \min\left\{\overline{b}_k, C - \underline{a}_k\right\}, \tag{II$'$}$$

$$\sum_{j\in N\setminus\{k\}} \left(\underline{b}_j + v_j\right) \le \min\left\{\overline{a}_k, C - \underline{b}_k\right\}, \tag{III$'$}$$

$$0 \le u_j \le \overline{a}_j - \underline{a}_j, \quad j \in N\setminus\{k\}, \tag{IV$'$}$$

$$0 \le v_j \le \overline{b}_j - \underline{b}_j, \quad j \in N\setminus\{k\}, \tag{V$'$}$$

where the objective function coefficients are given by

$$\alpha'_j = \alpha_j + \beta_k, \qquad \beta'_j = \beta_j + \alpha_k, \quad j \in N\setminus\{k\}. \tag{8}$$

Having found the solution to $LP'(AB)$, the optimal values of u_k and v_k are determined from the formulae:

$$u_k = \sum_{j\in N\setminus\{k\}} \left(\underline{b}_j + v_j\right) - \underline{a}_k, \tag{9}$$

$$v_k = \sum_{j\in N\setminus\{k\}} \left(\underline{a}_j + u_j\right) - \underline{b}_k. \tag{10}$$

Theorem 3. *Linear programming problem $LP(AB)$ combined with (7) is equivalent to the linear programming problem $LP'(AB)$ combined with (9) and (10).*

Proof. It is easy to verify that the objective function coefficients α'_j and β'_j of problem $LP'(AB)$ are obtained by substituting (9) and (10) in the objective function of problem $LP(AB)$ and removing the constant terms.

To show that the constraints of the two problems are equivalent we prove that (I)–(V) together with (7) imply (I$'$)–(V$'$) together with (9) and (10) and vice versa.

1. (I)–(V) and (7) imply (I$'$)–(V$'$) and (9)–(10).

Constraint (9) can be obtained by subtracting (7) from (II), while (10) can be obtained by subtracting (7) from (I).
Constraint (I$'$) is the sum of (I) and (II) minus (7).
Constraint (II$'$) follows from the following two relations:

- the difference of (I) minus (7) combined with the upper bound $v_k \le \overline{b}_k - \underline{b}_k$ from (V),
- equality (I) combined with the lower bound $u_k \ge 0$ from (IV).

Similarly, constraint (III$'$) follows from the following two relations:

- the difference of (II) minus (7) combined with the upper bound $u_k \le \overline{a}_k - \underline{a}_k$ from (IV),
- equality (II) combined with the lower bound $v_k \ge 0$ from (V).

Box inequalities (IV$'$), (V$'$) are the same as (IV), (V) for $j \in N\setminus\{k\}$.

2. (I$'$)–(V$'$) and (9)–(10) imply (I)–(V) and (7).

Constraint (7) can be obtained by adding (I$'$), (9) and (10).
Constraint (I) is the sum of (I$'$) and (9), while (II) is the sum of (I$'$) and (10).
For $j \in N\setminus\{k\}$, inequality (III) is dominated by (I$'$).
For $j = k$, inequality (III) reduces to (7).
For $j \in N\setminus\{k\}$, box inequalities (IV)–(V) and (IV$'$)–(V$'$) are the same.
For $j = k$, the lower bound of (IV) follows from (9) + (I$'$) combined with (II$'$) and the upper bound of (IV) follows from (9) and (III$'$).
Finally, for $j = k$, the lower bound of (V) follows from (10) + (I$'$) combined with (III$'$) and the upper bound of (V) follows from (10) and (II$'$).

Theorem 3 is proved. ■

Problem $LP'(AB)$ is known as the continuous *generalized upper bound resource allocation problem*, which is solvable in $O(n)$ time [8]. Thus our original problem $LP(AB)$ reduces to solving $LP(A)$ and $LP(B)$ and, if necessary, $LP'(AB)$. Summarizing, we can formulate the solution algorithm as follows.

Algorithm 'Optimum Processing Times'

Step 1. Find solutions $\mathbf{u}^A$, $\mathbf{v}^B$ to problems $LP(A)$, $LP(B)$.
Step 2. **If** $\mathbf{u}^A$ and $\mathbf{v}^B$ satisfy constraint (III), **then return** $\mathbf{u}^* = \mathbf{u}^A$, $\mathbf{v}^* = \mathbf{v}^B$.
Step 3. **Else** determine a critical job k, formulate $LP'(AB)$ and find its solution u'_j, v'_j, $j \in N \setminus \{k\}$. Use (9) and (10) to obtain u'_k, v'_k and **return** $\mathbf{u}^* = \mathbf{u}'$, $\mathbf{v}^* = \mathbf{v}'$.

Since Steps 1 and 3 involve two $O(n)$ algorithms from [1,8] and Step 2 verifies $n+1$ inequalities, the running time of the above algorithm is $O(n)$.

3. The properties of an optimal schedule

In this section we study the properties of an optimal schedule for problem $O2|\text{contr}, C_{\max} \le C|K$. Based on these properties, we design an efficient algorithm for solving the bicriteria problem $O2|\text{contr}|\,(C_{\max}, K)$ in the following section.

It is known [8] that the constraints of problems $LP(A)$, $LP(B)$ and $LP'(AB)$ are submodular and each problem can be solved by a greedy algorithm (see, e.g., [5]). The algorithm starts with zero values of all the variables u_j, v_j, $j \in N$. In each iteration, it selects the 'most profitable variable' that corresponds to the largest coefficient in the objective function and increases it by the maximum amount so that none of the constraints is violated. The process continues with the variable corresponding to the next largest coefficient, etc.

We distinguish between two types of optimal schedules depending on whether or not a critical job exists. A schedule of Type 1 does not possess a critical job and can be found by solving problems $LP(A)$ and $LP(B)$; a schedule of Type 2 has a critical job and can be found by solving problem $LP'(AB)$.

Consider an optimal schedule of Type 1. Problem $LP(A)$ deals with the operations on machine A. In accordance with the greedy approach, they are considered in the order

$$\alpha_{i_1} \ge \alpha_{i_2} \ge \cdots \ge \alpha_{i_n} \ge \alpha_{i_{n+1}} = 0, \tag{11}$$

and are decompressed one by one by the maximum amount without violating any of the constraints of $LP(A)$.

Similarly, for $LP(B)$, the operations on machine B are considered in the order

$$\beta_{j_1} \ge \beta_{j_2} \ge \cdots \ge \beta_{j_n} \ge \beta_{j_{n+1}} = 0, \tag{12}$$

and are decompressed one by one by the maximum amount without violating any of the constraints of $LP(B)$.

In the resulting schedule, the operations with the maximum compression costs on machines A and B are fully decompressed, the operations with the smallest costs are fully compressed, and on each machine one operation at most is partially compressed. Using the sequences (11) and (12) for machines A and B, respectively, we specify the compressed and decompressed operations as follows:

Type 1 *schedule*

Machine A: $\underbrace{\boxed{i_1, i_2, \ldots, i_{\ell-1}}}_{\text{decompressed}}$, $\underbrace{i_\ell}_{\substack{\text{(partially)}\\ \text{compressed}}}$, $\underbrace{i_{\ell+1}, \ldots, i_n, i_{n+1}}_{\text{compressed}}$,

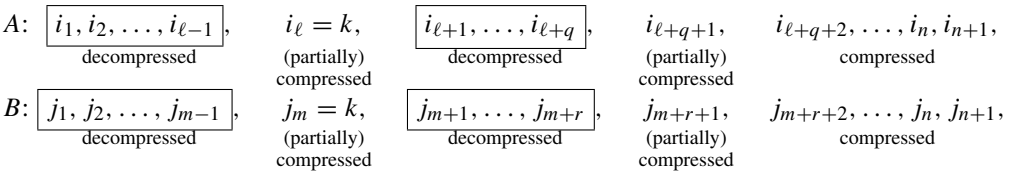
Machine B: $\underbrace{\boxed{j_1, j_2, \ldots, j_{m-1}}}_{\text{decompressed}}$, $\underbrace{j_m}_{\substack{\text{(partially)}\\ \text{compressed}}}$, $\underbrace{j_{m+1}, \ldots, j_n, j_{n+1}}_{\text{compressed}}$,

where $\ell, m \ge 1$. Observe that if $\ell = 1$ or $m = 1$, then the set of fully decompressed operations is empty for the corresponding machine.

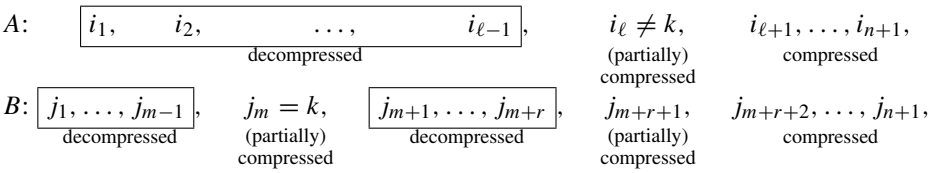
Consider now an optimal schedule of Type 2 corresponding to problem $LP'(AB)$ to which we pass when a combination of the optimal solutions to problems $LP(A)$ and $LP(B)$ provides an infeasible solution to $LP(AB)$ with a critical job k. As with the greedy approach, the largest decompression amounts are found by considering operations

of the jobs from $N \setminus \{k\}$ one by one in non-increasing order of the coefficients α'_j and β'_j. Due to (8), in the resulting schedule, the jobs from the set $N \setminus \{k\}$ with the maximum costs α_j and β_j are fully decompressed, the jobs with the smallest costs are fully compressed, and on each machine one job at most from $N \setminus \{k\}$ is partially compressed. Observe that the optimal decompression amounts of the two operations of job k are determined from (9)–(10) after the greedy algorithm terminates. As with Corollary 2, the two operations of job k cannot be fully compressed or fully decompressed simultaneously. Hence, the three possible cases representing the structure of an optimal schedule of Type 2 can be described as follows:

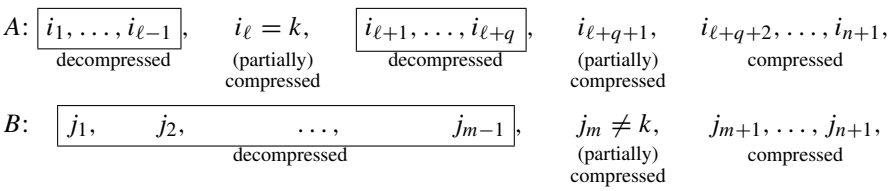
Type 2(a) schedule



Type 2(b) schedule



Type 2(c) schedule



where $\ell, m \geq 1$ and $q, r \geq 0$. Observe that if $q = r = 0$, then a schedule of Type 2(a) has a structure similar to that of Type 1. In a schedule of Type 2(b), the operation of job k on machine A is either fully compressed or fully decompressed. Similarly, in a schedule of Type 2(c), the operation of job k on machine B is either fully compressed or fully decompressed.

We assume that in the schedule of Type 2(b) $i_\ell \neq k$; otherwise it is equivalent to a schedule of Type 2(a) with $q = 0$. Similarly, in the schedule of Type 2(c) $j_m \neq k$; otherwise it is equivalent to a schedule of Type 2(a) with $r = 0$.

4. Bicriteria problem

Consider the bicriteria problem $O2|\text{contr}|\,(C_{\max}, K)$ of minimizing $C_{\max}$ and K simultaneously. A solution to a bicriteria problem is the set of Pareto optimal points in the (C, K)-space, where $C = C_{\max}$ denotes the makespan and K is the compression cost function (1). A schedule S' is called Pareto optimal if there exists no schedule S'' such that $C(S'') \leq C(S')$ and $K(S'') \leq K(S')$, where at least one of these relations holds as a strict inequality. The set of Pareto optimal points is given by the break-points of the so-called efficient frontier; see [17] for definitions and a state-of-the-art survey of multicriteria scheduling.

We start with compressing all operations to their minimum processing times $\underline{a}_j$ and $\underline{b}_j$, $j \in N$, and finding the first Pareto optimal point $\left(C^0, K^0\right)$ with the smallest makespan $C^0 = \underline{C}$ given by (2). In general, in an optimal schedule some jobs can be decompressed without increasing the makespan. The optimal decompression amounts u_j, v_j that minimize the compression cost K can be found in $O(n)$ time by applying algorithm 'Optimum Processing Times' from Section 2 to a single-criterion problem $O2|\text{contr}, C_{\max} \leq C^0|K$. Denote the found schedule by S^0, optimal decompression amounts by u_j^0 and v_j^0, $j \in N$, and determine the compression cost $K^0 = \sum_{j \in N} \left(\alpha_j \left(\overline{a}_j - \underline{a}_j - u_j^0\right) + \beta_j \left(\overline{b}_j - \underline{b}_j - v_j^0\right)\right)$. Observe that the structure of that schedule can be of Type 1 or Type 2 (a–c).

Consider now the problem $O2|\text{contr}, C_{\max} \le C'|K$ with a larger makespan value $C' > C^0$. Let S' be the corresponding optimal schedule. Both schedules S' and S^0 can be constructed by a greedy approach described in Section 3 so that decompression amounts of all the operations in S' are no smaller than those in S^0 and at least one pair of operations i, j is decompressed further on machines A and B, respectively. Depending on the type of schedule S^0, this pair of operations is given by

$$(i, j) = \begin{cases} (i_\ell, j_m), & \text{if } S^0 \text{ is of Type 1, 2(b) or 2(c),} \\ (i_{\ell+q+1}, j_m), & \text{if } S^0 \text{ is of Type 2(a) and } \alpha'_{i_{\ell+q+1}} \ge \beta'_{j_{m+r+1}}, \\ (i_\ell, j_{m+r+1}), & \text{if } S^0 \text{ is of Type 2(a) and } \alpha'_{i_{\ell+q+1}} < \beta'_{j_{m+r+1}}, \end{cases} \tag{13}$$

where $\alpha'_{i_{\ell+q+1}}$ and $\beta'_{j_{m+r+1}}$ represent the combined costs defined by (8):

$$\alpha'_{i_{\ell+q+1}} = \alpha_{i_{\ell+q+1}} + \beta_{j_m},$$
$$\beta'_{j_{m+r+1}} = \beta_{j_{m+r+1}} + \alpha_{i_\ell}.$$

It follows that given the initial Pareto optimal point (C^0, K^0), the next break-point of the efficient frontier (C^1, K^1) can be found by decompressing the operations i and j specified by (13) on machines A and B. Due to (3), in both schedules S^0 and S^1 machines A and B are permanently busy in the time intervals $[0, C^0]$ and $[0, C^1]$, respectively, so that a transition from S^0 to S^1 is performed by decompressing operations i and j by the same amount z:

$$u_i^1 = u_i^0 + z,$$
$$v_j^1 = v_j^0 + z,$$

with the makespan increasing by z:

$$C^1 = C^0 + z.$$

This guarantees that for schedule S^1 condition (3) remains satisfied.

In what follows we consider problem $LP(AB)$ and determine the largest possible decompression amount z such that constraints (I)–(V) are observed. Note that due to the equivalence of problems $LP(AB)$ and $LP'(AB)$ in the presence of a critical job (which is characterized by (7) or (9)–(10)), the same formula for z guarantees that the constraints of problem $LP'(AB)$ are satisfied as well.

Constraints (I)–(II) hold for S^1 due to the observation that (3) remains satisfied.

Constraints (IV)–(V) are satisfied if

$$0 \le u_i^0 + z \le \overline{a}_i - \underline{a}_i,$$
$$0 \le v_j^0 + z \le \overline{b}_j - \underline{b}_j,$$

or equivalently,

$$z \le \min\left\{\overline{a}_i - \left(\underline{a}_i + u_i^0\right), \overline{b}_j - \left(\underline{b}_j + v_j^0\right)\right\}.$$

The decompression of operations i, j does not cause violation of constraint (III) if these operations correspond to different jobs. Otherwise, e.g., if $i = j$, only one constraint from (III) is affected, namely

$$\left(\underline{a}_i + u_i^0 + z\right) + \left(\underline{b}_i + v_i^0 + z\right) \le C^0 + z,$$

which gives an additional upper bound on the decompression amount z:

$$z \le C^0 - \left(\left(\underline{a}_i + u_i^0\right) + \left(\underline{b}_i + v_i^0\right)\right).$$

Summarizing, we conclude that as the processing times of operations i and j grow, the structure of the corresponding schedule may change, and the situation that such a change takes place determines the next break-point

of the efficient frontier. A possible structural change occurs for

$$z = \begin{cases} \min\left\{\overline{a}_i - \left(\underline{a}_i + u_i^0\right), \overline{b}_j - \left(\underline{b}_j + v_j^0\right)\right\}, & \text{if } i \neq j, \\ \min\left\{\overline{a}_i - \left(\underline{a}_i + u_i^0\right), \overline{b}_i - \left(\underline{b}_i + v_i^0\right), C^0 - \left(\left(\underline{a}_i + u_i^0\right) + \left(\underline{b}_i + v_i^0\right)\right)\right\}, & \text{if } i = j, \end{cases} \tag{14}$$

and it is associated with one of the following three events:

Event 1: an operation of a non-critical job becomes fully decompressed;
Event 2: an operation of a critical job becomes fully decompressed;
Event 3: in a schedule without a critical job, a decompressible job becomes critical.

Event 1 may happen for a schedule of any type. As a result, the schedule type does not change but a new operation replaces i or j for further decompression.

Event 2 may happen for any Type 2 schedule, so that a schedule of Type 2(a) is transformed into Type 2(b) or 2(c), while a schedule of Type 2(b) or 2(c) is transformed into Type 1.

Finally, Event 3 may occur only for a schedule of Type 1 if the operations of the same job $i = j$ are being decompressed. The resulting schedule can be of Type 2(a), 2(b) or 2(c).

This structural change corresponds to a new break-point $\left(C^1, K^1\right)$ for which a pair of decompressible operations i, j should be updated. Since the structure of the resulting schedule S^1 is of Type 1 or Type 2, we can proceed in a similar way constructing the subsequent break-points $\left(C^2, K^2\right), \ldots, (C^g, K^g)$ that satisfy

$$C^0 < C^1 < C^2 < \cdots < C^g,$$
$$K^0 > K^1 > K^2 > \cdots > K^g = 0.$$

The algorithm that performs the transition from one break-point to another can be formulated as follows.

Algorithm 'Transition from (C^h, K^h) **to** (C^{h+1}, K^{h+1})'

Given: Pareto optimal point $\left(C^h, K^h\right)$ and the corresponding schedule S^h of Type 1 or Type 2 with actual processing times $a_j, b_j, j \in N$.

Step 1. Select a pair of jobs i, j to be decompressed on machines A, B in accordance with (13) considering schedule S^h instead of S^0.

Step 2. Determine the decompression amount z

$$z = \begin{cases} \min\left\{\overline{a}_i - a_i, \overline{b}_j - b_j\right\}, & \text{if } i \neq j, \\ \min\left\{\overline{a}_i - a_i, \overline{b}_i - b_i, C^h - (a_i + b_i)\right\}, & \text{otherwise}, \end{cases}$$

and decompress job i on machine A and job j on machine B:

$$a_i := a_i + z, \qquad b_j := b_j + z.$$

Step 3. Set $C^{h+1} := C^h + z$, $K^{h+1} := K^h - \left(\alpha_i + \beta_j\right) z$, $h := h+1$ and update i_ℓ, $i_{\ell+q+1}$, j_m and j_{m+r+1}, if required.

All the steps of the algorithm require $O(1)$ time, given the job orders (11) and (12). Each time one operation on machine A or B becomes fully decompressed or one of the n jobs becomes critical. Thus the total number of break-points does not exceed $3n + 1$. Taking into account that job orders (11) and (12) can be found in $O(n \log n)$ time, the overall time complexity of constructing all the break-points of the efficient frontier is $O(n \log n)$.

5. Conclusions

We presented the $O(n)$-time algorithm for solving the single criterion open shop problem with controllable processing times and the $O(n \log n)$-time algorithm for its bicriteria counterpart.

The single criterion problem $O2|\text{contr}, C_{\max} \leq d| \sum \left(\alpha_j x_j + \beta_j y_j\right)$ is closely related to the late-work problem $O2\| \sum(\alpha_j X_j + \beta_j Y_j)$ studied in [2]. In the latter problem, the jobs have fixed (uncontrollable) processing times and a common due date d. The late work of a job j is defined as the amount of work X_j and Y_j executed on the two machines after the due date d. The objective is to minimize the total late work $\sum_{j=1}^{n}(\alpha_j X_j + \beta_j Y_j)$. Observe that in

the late-work model [2] the late parts X_j and Y_j of the two operations of job j are included in a schedule, while in the model with controllable processing times studied in the current paper the two operations of job j are compressed by the amounts x_j and y_j so that the corresponding parts of lengths x_j and y_j are not included in a schedule.

It was shown in [2] that in the case of unit penalties $\alpha_j = \beta_j = 1$, $j = 1, \ldots, n$, the late-work problem is solvable in $O(n)$ time, while in the case of arbitrary penalties it is NP-hard.

Another relevant problem is the open shop problem with controllable machine speeds, for which varying the speed of a machine causes the processing times of all operations on that machine to change by the same factor. As shown in [19], the latter problem is solvable in $O(n)$ time.

Observe that in the case of a flow shop, the two-machine problem with controllable processing times is NP-hard [9, 16]; its counterpart with total late work is NP-hard as well, even in the case of unit penalties $\alpha_j = \beta_j = 1$, while the two-machine problem with controllable machine speeds is solvable in $O(n \log n)$ time [18].

Acknowledgements

The authors are grateful to anonymous referees for their constructive suggestions, which led to a better proof of Theorem 1 and improved presentation of the results.

T.C.E. Cheng was supported in part by The Hong Kong Polytechnic University under a grant from the Area of Strategic Development in China Business Services.

References

[1] E. Balas, E. Zemel, An algorithm for large zero-one knapsack problems, Oper. Res. 28 (1980) 1130–1154.
[2] J. Błażewicz, E. Pesch, M. Sterna, F. Werner, Open shop scheduling problems with late work criteria, Discrete Appl. Math. 134 (2004) 1–24.
[3] B. Chen, C.N. Potts, G.J. Woeginger, A review of machine scheduling: Complexity, algorithms and approximability, in: D.-Z. Du, P.M. Pardalos (Eds.), Handbook of Combinatorial Optimization, Kluwer Academic Publishers, 1998, pp. 21–169.
[4] R.L. Daniels, J.B. Mazzola, Flow shop scheduling with resource flexibility, Oper. Res. 42 (1994) 504–522.
[5] A. Frank, E. Tardos, Generalized polymatroids and submodular flows, Math. Program. 42 (1988) 489–563.
[6] T. Gonzalez, S. Sahni, Open shop scheduling to minimize finish time, J. ACM 23 (1976) 665–679.
[7] J. Grabowski, A. Janiak, Job-shop scheduling with resource-time models of operations, European J. Oper. Res. 28 (1986) 58–73.
[8] D.S. Hochbaum, S.-P. Hong, About strongly polynomial time algorithms for quadratic optimization over submodular constraints, Math. Program. 69 (1995) 269–309.
[9] A. Janiak, Minimization of resource consumption under a given deadline in the two-processor flow-shop scheduling problem, Inform. Process. Lett. 32 (1989) 101–112.
[10] A. Janiak, Minimization of the makespan in a two-machine problem under given resource constraints, European J. Oper. Res. 98 (1998) 325–337.
[11] A. Janiak, M.-C. Portmann, Genetic algorithm for the permutation flow-shop scheduling problem with linear models of operations, Ann. Oper. Res. 83 (1998) 95–114.
[12] K. Jansen, M. Mastrolilli, R. Solis-Oba, Approximation schemes for job shop scheduling problems with controllable processing times, European J. Oper. Res. 167 (2005) 297–319.
[13] E.L. Lawler, J.K. Lenstra, A.H.G. Rinnooy Kan, D.B. Shmoys, Sequencing and scheduling: Algorithms and complexity, in: S.C. Graves, A.H.G. Rinnooy Kan, P.H. Zipkin (Eds.), Handbooks in Operations Research and Management Science, in: Logistics of Production and Inventory, vol. 4, North-Holland, Amsterdam, 1993, pp. 455–522.
[14] E. Nowicki, An approximation algorithm for the m-machine permutation flow-shop scheduling problem with controllable processing times, European J. Oper. Res. 70 (1993) 342–349.
[15] E. Nowicki, S. Zdrzalka, Two-machine flow shop scheduling problem with controllable job processing times, European J. Oper. Res. 34 (1988) 208–220.
[16] E. Nowicki, S. Zdrzalka, A survey of results for sequencing problems with controllable processing times, Discrete Appl. Math. 26 (1990) 271–287.
[17] V. T'kindt, J.-C. Billaut, Multicriteria Scheduling: Theory, Models and Algorithms, Springer, Berlin, 2002.
[18] C.P.M. Van Hoesel, A.P.M. Wagelmans, M. Van Vliet, An $O(n \log n)$ algorithm for the two-machine flow shop problem with controllable machine speeds, INFORMS J. Comput. 8 (1996) 376–382.
[19] M. Van Vliet, Optimization of manufacturing system design, Ph.D. Thesis, Econometric Institute, Erasmus University Rotterdam, The Netherlands, 1991.